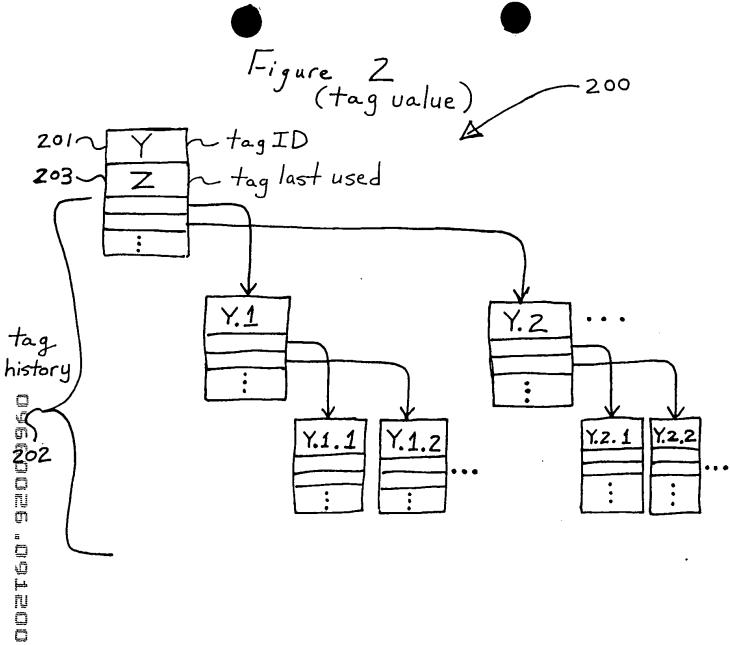
Liqure 1 HDL -100 Stimulus Source 101 Vectors Code sub-files: · Expressions Instrumentation · Instrumented Instrumen tation to Non-Process Mapping _instrumented File 5 111 Instrumented 103 HDL Source 106 Code 104 105 Monitoring HDL Validation PLI Test Simulation Process Results Process 110 sub-parts: 109 · Observability Coverage Reporting Process · Blocked Observability Observability History Tags Report File 108 107

¥



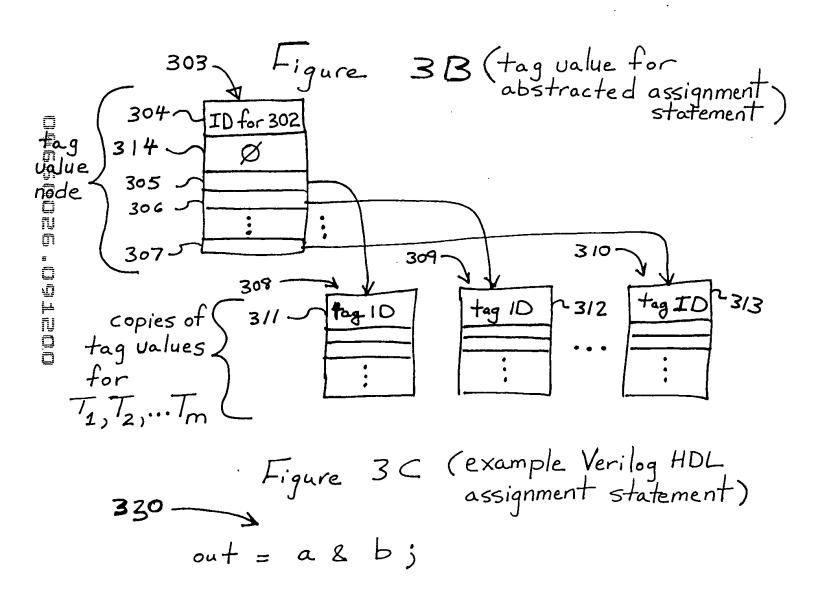


Figure 3D

(example Verilog HDL assignment statement)

out_1 = $(in_3 \& in_2) || (in_1 \& in_0)$

Figure 3E (parse tree of rhs of assignment statement with rule-based tag propagation)

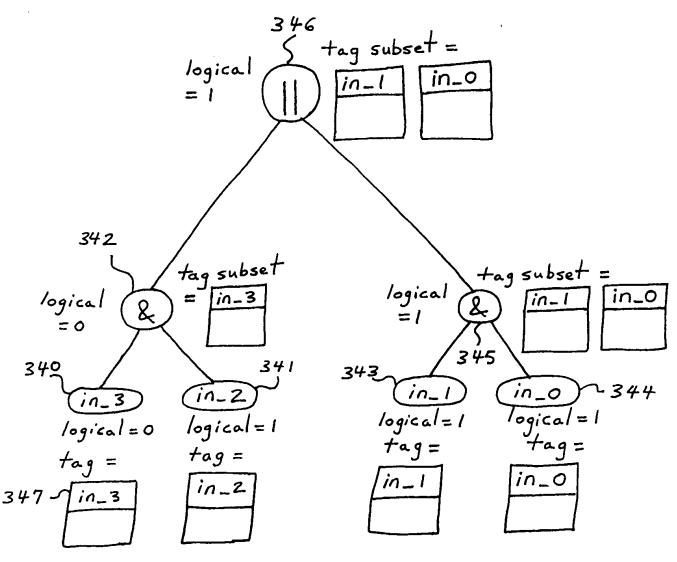


Figure 3F (resulting tag value from rule-based propagation)

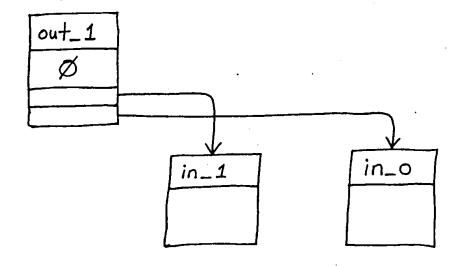


Figure 4 (abstracted conditionally executed assignment)

$$\begin{array}{c}
\downarrow \text{ if } (f_1(W_1, W_2, ... W_t)) \\
\downarrow \text{ if } (f_2(Z_1, Z_2, ... Z_p)) \\
\downarrow \text{ out } = f(x_1, x_2, ... x_n); \\
\downarrow \text{ 300} \\
\downarrow \text{ 300}
\end{array}$$

Figure 5A

(Verilog HDL program fragment with asynchronous assignments, an atomic block of synchronous assignments and conditionally executed assignments)

```
10
        assign x = a \& b;
20
        assign y = a \parallel c;
        aways @(posedge clk)
        begin
30
               e = a \& d;
               f = b || h;
40
               if (a & (c | d))
                       if (x || y)
50
                               e = f \& g;
                       else
60
                               e = f \parallel g;
        end
```

Figure 5B

(instrumented Verilog HDL program fragment)

```
10
       assign x = a \& b;
20
       assign y = a || c;
       aways @(posedge clk)
       begin
30
               $pli (30);
40
               e = a \& d;
50
              f = b || h;
              if ( a & (c || d) )
                      if (x || y)
                             begin
60
                                     $pli(60);
70
                                     e = f \& g;
                             end
                      else
                             begin
80
                                     $pli(80);
90
                                     e = f \parallel g;
                             end
       end
```

Figure 5C

Instrumented to Non-instrumented sub-file

- 10 10
- 20 20
- 30 40
- 40 50
- 50 70
- 60 90

Figure 5D

Expressions sub-file

PLI_call_instrumented_line_num 30 assign_stmt_instrumented_line_num 40 lhs_signal "e" rhs_expression "a & d"

PLI_call_instrumented_line_num 30 assign_stmt_instrumented_line_num 50 lhs_signal "f" rhs_expression "b || h"

PLI_call_instrumented_line_num 60 assign_stmt_instrumented_line_num 70 lhs_signal "e" rhs_expression "f & g", "a&(c||d)", "x||y"

PLI_call_instrumented_line_num 80 assign_stmt_instrumented_line_num 90 lhs_signal "e" rhs_expression "f || g", "a&(c||d)", "x||y"

PLI_call_instrumented_line_num 0 assign_stmt_instrumented_line_num 10 lhs_signal "x" rhs_expression "a&b"

PLI_call_instrumented_line_num 0 assign_stmt_instrumented_line_num 20 lhs_signal "y" rhs expression "a || c"

Figure 5F (hierarchical Verilog HDL program fragment)

```
/* Higher-level module */
module higher_mod;

10 lower_mod lower_mod_inst(z, x, y & q)
endmodule

/* Lower-level modue */
module lower_mod (out1, in1, in2);
assign out1 = in1 || in2;
endmodule
```

Figure 5G

```
Expressions sub-file

PLI_call_instrumented_line_num 0

assign_stmt_instrumented_line_num 0

lhs_signal "lower_mod_inst.in2"

rhs_expression "y & q"
```

Figure 5H

```
parameter passed to callback function when signal "y" changes PLI_call_instrumented_line_num 0 assign_stmt_instrumented_line_num 0 lhs_signal "lower_mod_inst.in2" rhs_expression "y & q"

parameter passed to callback function when signal "q" changes PLI_call_instrumented_line_num 0 assign_stmt_instrumented_line_num 0
```

parameter passed to callback function when signal "x" changes PLI_call_instrumented_line_num 0 assign_stmt_instrumented_line_num 0 lhs_signal "lower_mod_inst.in1" rhs_expression "x"

parameter passed to callback function when signal "lower_mod_inst.out1" changes

PLI_call_instrumented_line_num 0 assign_stmt_instrumented_line_num 0 lhs_signal "z" rhs expression "lower mod inst.out1"

lhs signal "lower mod inst.in2"

rhs_expression "y & q"

FIG. 6

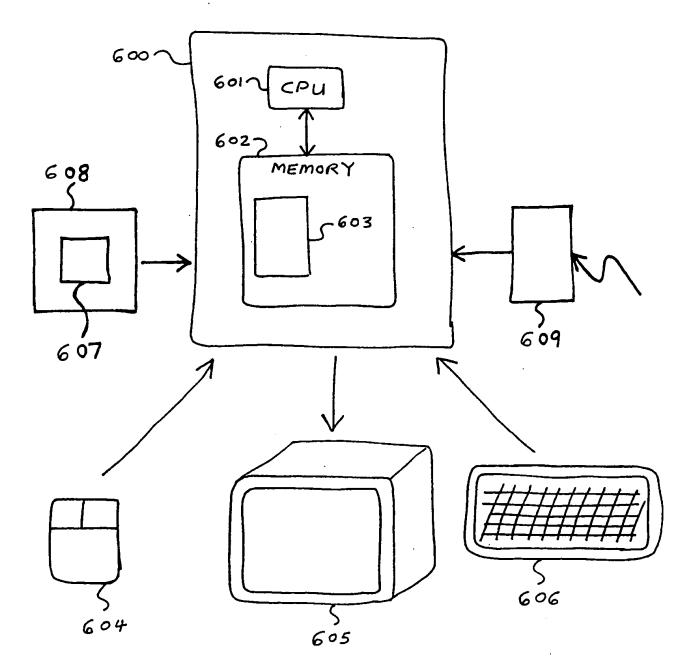




Figure 5E

```
parameter passed to callback function when signal "a" changes
PLI_call_instrumented_line_num 0
assign_stmt_instrumented_line_num 10
lhs_signal "x"
rhs_expression "a&b"

PLI_call_instrumented_line_num 0
assign_stmt_instrumented_line_num 20
lhs_signal "y"
rhs_expression "a || c"
```

parameter passed to callback function when signal "b" changes PLI_call_instrumented_line_num 0 assign_stmt_instrumented_line_num 10 lhs_signal "x" rhs_expression "a&b"

parameter passed to callback function when signal "c" changes
PLI_call_instrumented_line_num 0
assign_stmt_instrumented_line_num 20
lhs_signal "y"
rhs_expression "a || c"